**SAMUEL AJI SENA, 0610630097**
Jurusan Teknik Elektro, Fakultas Teknik Universitas Brawijaya, 2013
**Perancangan dan pembuatan *Application Programming Interface Server* untuk Arduino**
Dosen Pembimbing :
**Adharul Muttaqin, ST., MT & R. Arief Setyawan, ST., MT**

*Abstrak* **- Arduino adalah sebuah platform untuk melakukan komputasi fisis berbasis mikrokontroler. Beberapa sensor dapat dihubungkan pada input arduino dan aktuator dapat dihubungkan pada output sehingga membentuk suatu sistem. Saat ini arduino sudah banyak digunakan sebagai *remote system* melalui jaringan komputer namun karena keterbatasan sumber daya yang dimiliki arduino akan sangat sulit untuk membuat sebuah sistem yang optimal.**

**Skripsi ini membahas perancangan dan pembuatan *application programming interface server* yang berfungsi sebagai jembatan antara aplikasi dan arduino pada jaringan komputer atau internet sehingga dapat memudahkan pemrogram untuk membuat aplikasi dan meringankan kerja arduino. Protokol yang digunakan oleh aplikasi untuk berkomunikasi dengan *server* adalah HTTP (*Hypertext Transfer Protocol*) sedangkan protokol yang digunakan oleh server untuk berkomunikasi dengan arduino adalah TCP (*Transmission Control Protocol*). Fungsi-fungsi yang dapat dilakukan oleh *server* adalah fungsi-fungsi input dan output.**

**Dari hasil pengujian semua fungsi input dan output dapat dilakukan. Pada pengujian dengan menggunakan satu buah request, total waktu rata-rata yang dibutuhkan untuk melakukan satu operasi adalah 14,8 ms. Dalam hal ini sistem dapat berjalan dengan baik dan memiliki performa yang cukup bagus karena delay sistem tidak akan dirasakan pengaruhnya oleh client.**

## I. Pendahuluan

### 1.1. Latar Belakang

Perkembangan teknologi di bidang elektronika pada era ini sangat pesat. Saat ini banyak bermunculan *electronic board* yang memiliki fitur yang berbeda-beda, salah satunya adalah arduino. Arduino adalah *platform* untuk melakukan komputasi fisis yang berbasis mikrokontroler. Arduino dapat merasakan lingkungan sekitar dengan cara menghubungkan berbagai jenis sensor pada input dan dapat mengendalikan sesuatu dengan cara menghubungkan aktuator pada output.

Untuk membuat suatu aplikasi *client-server* yang berjalan pada jaringan komputer dan dapat berinteraksi dengan arduino, aplikasi tersebut harus mampu membentuk koneksi dan melakukan komunikasi data dengan arduino sedangkan arduino juga harus dapat melakukan hal tersebut. Dengan semakin berkembangnya teknologi, aplikasi yang dapat dibuat tidak hanya aplikasi *desktop* saja tetapi aplikasi berbasis *web* maupun aplikasi *mobile*. Agar dapat membentuk koneksi dan melakukan komunikasi data, aplikasi dan arduino harus menggunakan protokol yang sama. Dengan keterbatasan sumber daya yang dimiliki oleh arduino, akan sangat sulit untuk mengimplementasikan sistem tersebut, sehingga diperlukan sebuah interface server yang dapat menjembatani aplikasi dan arduino.

### 1.2 Ruang Lingkup

Ruang lingkup penulisan tugas akhir ini akan dibatasi pada.

1. Arduino yang digunakan adalah Arduino Uno R3.
2. Ethernet *shield* yang digunakan adalah berbasis chip WIZNET W5100.
3. Menggunakan *Hypertext Transfer Protocol* (HTTP) sebagai protokol antara aplikasi dengan *Application Programming Interface Server* mengacu pada RFC 2616.
4. Menggunakan *Transmission Control Protocol* (TCP) sebagai protokol *transport* antara *Application Programming Interface Server* dengan Arduino mengacu pada RFC 793.
5. *Application Programming Interface Server* dibuat dengan menggunakan bahasa skrip PHP.

## II. Tinjauan Pustaka

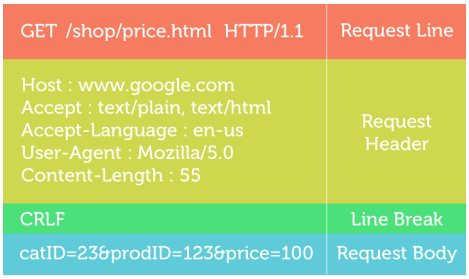
### 2.1 Protokol TCP/IP

TCP/IP (*Transmission Control Protocol/Internet Protocol*) merupakan sekelompok protokol yang mengatur komunikasi data komputer di internet. Karena menggunakan protokol yang sama, maka perbedaan jenis komputer dan sistem operasi antara dua buah komputer tidak menjadi masalah. TCP/IP terdiri dari empat *layer*, yaitu

- *Network Access/Interface Layer*
- *Internet Layer*
- *Host-to-Host Transport Layer*
- *Application Layer*

### 2.1.1. HTTP (*Hypertext Transfer Protocol*)

HTTP merupakan protokol yang terletak di *application layer* pada arsitektur protokol TCP/IP. HTTP tidak menentukan bagaimana data diproses tetapi HTTP bertanggung jawab menentukan bagaimana data ditransfer. HTTP tidak hanya dapat mentransfer halaman web saja tetapi dapat digunakan untuk mentransfer semua format data, bukan hanya web *browser* saja yang dapat menggunakan protokol ini, tetapi aplikasi lain juga dapat melakukan transfer data dengan menggunakan protokol HTTP.



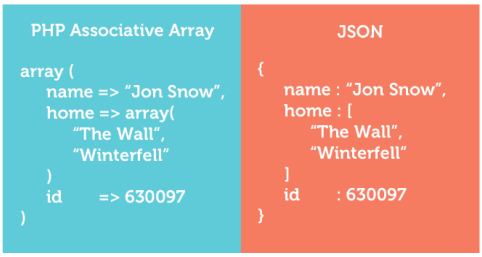

**Gambar 2.1. Format HTTP Request**
**Sumber: Mansfield, 2004:553**

### 2.2 *Application Programming Interface Server*

*Appication Programming Interface* (API) adalah sekumpulan fungsi, perintah dan protokol yang dapat digunakan untuk menghubungkan satu aplikasi dengan aplikasi yang lain agar dapat berinteraksi. Seiring dengan perkembangan internet, API dapat diimplementasikan pada sisi server dan dapat digunakan oleh beberapa aplikasi yang dapat terhubung ke server dengan menggunakan protokol tertentu. Pada protokol HTTP, *Appication Programming Interface* umumnya disebut sebagai *Web Appication Programming Interface Server* atau *Web Service.*

### 2.2.1 JSON (Javascript Object Notation)

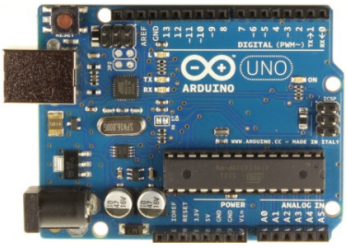
JSON adalah sebuah berkas yang umumnya digunakan sebagai pertukaran data pada internet. Berkas JSON berbasis teks sehingga mudah dikenali oleh berbagai macam bahasa pemrograman sehingga sangat ideal untuk digunakan dalam pertukaran data antar aplikasi yang berbeda bahasa pemrogramannya.




**Gambar 2.2. Format JSON**

### 2.3 Arduino

Arduino adalah *platform* untuk melakukan komputasi fisis yang berbasis mikrokontroler. Arduino dapat merasakan lingkungan sekitar dengan cara menghubungkan berbagai jenis sensor pada input dan dapat mengendalikan sesuatu dengan cara menghubungkan aktuator pada output. Salah satu kelebihannya adalah arduino dapat dihubungkan dengan *board* yang lain atau biasa disebut *arduino shield* sehingga fungsi dari arduino tersebut dapat diperluas lagi.




**Gambar 2.3. Arduino**

### 2.3.1 Arduino UNO R3

Arduino memliki beberapa jenis yang fitur dan fungsinya berbeda antara satu dengan yang lainnya. Arduino UNO R3 adalah salah satu jenis dari arduino yang ada. Berikut ini adalah spesifikasi lengkap dari Arduino UNO R3 :

| Spesifikasi | Detail |
|---|---|
| Microcontroller | ATmega328 |
| Operating Voltage | 5V |
| Input Voltage | 7-12V |
| Input Voltage | 6-20V |
| Digital I/O Pins | 14 (6 PWM output) |
| Analog Input Pins | 6 |
| DC Current per I/O Pin | 40 mA |
| DC Current for 3.3V Pin | 50 mA |
| Flash Memory | 32 KB (ATmega328) of which 0.5 KB used by bootloader |
| SRAM | 2 KB (ATmega328) |

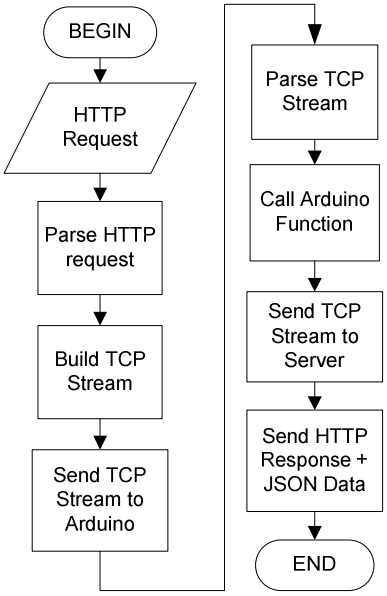
| | |
|---|---|
| EEPROM | 1 KB (ATmega328) |
| Clock Speed | 16 MHz |

**Tabel 2.1. Spesifikasi Arduino UNO R3**

## III. Perancangan

### 3.1 Perancangan Sistem

Perancangan sistem secara umum merupakan tahap awal sebagai acuan dalam perancangan sistem yang akan dibuat. Perancangan ini didahului dengan pendefinisian kegiatan client dalam menggunakan *Application Programming Interface Server* untuk arduino. Untuk memudahkan pemahaman, perancangan sistem secara keseluruhan dapat disajikan dalam diagram blok sistem berikut :




**Gambar 3.1. Diagram Alir Kerja Sistem**

#### 3.1.1 Perancangan metode read pin

Untuk melakukan proses read pada pin tertentu, HTTP request yang dikirimkan harus menggunakan method GET dengan pola Request-URI sebagai berikut :

```
/read/[mode]/[pin number]
```

**Gambar 3.2. Read Pin Request-URI**

#### 3.1.2 Perancangan metode write pin

Untuk melakukan proses write pada pin tertentu. HTTP request yang dikirimkan harus menggunakan method POST dengan Request-URI dan Entity POST sebagai berikut :

| | |
|---|---|
| Request-URI | /write |
| Entity POST | mode=[tipe pin]&pin=[nomor pin]&value=[nilai] |

**Gambar 3.3. Write Pin Request-URI dan Entity POST**

#### 3.1.3 Perancangan data JSON

Terdapat tiga macam data JSON yang akan dikirimkan ke client. Jika operasi *read* atau *write* berhasil, data JSON yang dibuat terdiri dari dua buah *object* yaitu status dan data. Jika operasi *read* atau *write* gagal, data JSON yang dibuat hanya terdiri dari satu *object* yang memiliki dua buah *member* yaitu result dan errormsg. Nilai dari member result adalah "failed" dan nilai dari errormsg bervariasi tergantung dari kesalahan yang terjadi.

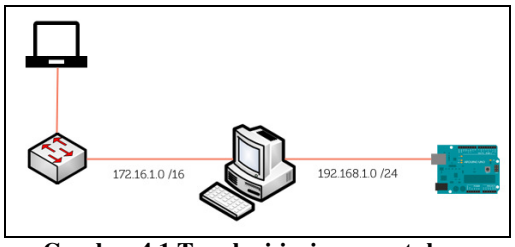
| READ | WRITE | ERROR |
|---|---|---|
| { status : { action: "read", result : "success" }, data : { mode : [tipe pin], pin : [nomor pin], value : [return value] } } | { status : { action: "write", result : "success" }, data : { mode : [tipe pin], pin : [nomor pin], value : [return value] } } | { status : { result : "failed", errormsg : [error] } } |

**Gambar 3.4. Struktur Data JSON**

## IV. Pengujian

### 4.1 Pengujian Sistem

Pengujian pada skripsi ini dibagi menjadi empat bagian yaitu pengujian operasi *read*, pengujian operasi *write*, pengujian *delay* untuk mengetahui waktu total proses dan pengujian *error* untuk mengetahui bagaimana sistem menangani kesalahan. Pengujian dilakukan dengan menggunakan satu buah komputer sebagai client yang berada pada *network address* 172.16.1.0 /16 sedangkan arduino berada pada *network address* 192.168.1.0 /24




**Gambar 4.1 Topologi jaringan untuk pengujian**

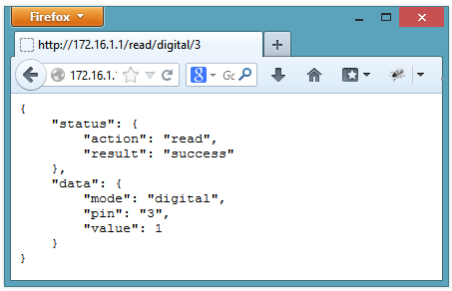
#### 4.1.1 Pengujian Operasi Read Pin

Pengujian operasi *read* dapat dikatakan berhasil jika *client* dapat membaca input pada arduino dengan benar. Pengujian dilakukan dengan konfigurasi sebagai berikut:

1. Menggunakan aplikasi *browser* Mozilla Firefox 21 untuk mengirimkan GET request ke server.
2. Pada pengujian digital *read*, pin yang digunakan adalah pin 3 dengan menggunakan pin 5V sebagai input.
3. Pada pengujian analog *read*, pin yang digunakan adalah pin A5 dengan menggunakan pin 5V dan pin 3,3V sebagai input.

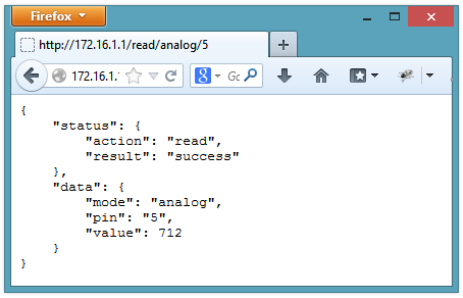
Hasil pengujian operasi *read* dapat dilihat sebagai berikut :

1. Digital Read (HIGH)




**Gambar 4.2 Hasil operasi digital read pada pin 3 (HIGH)**

2. Analog Read (3,3 volt)




**Gambar 4.3 Hasil operasi analog read pada pin A5 (3,3 volt)**

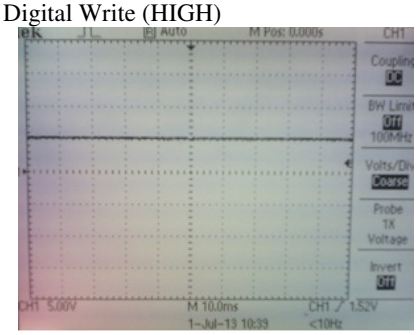
### 4.1.2 Pengujian Operasi Write Pin

Pengujian operasi write dapat dikatakan berhasil jika *client* dapat memberikan nilai pada pin output arduino dengan benar. Pengujian dibagi menjadi dua bagian yaitu *digitalWrite* dan *analogWrite*. Pengujian dilakukan di Laboratorium Elektronika dengan prosedur sebagai berikut:

1. Membuat sebuah script PHP dengan memanfaatkan modul cURL untuk mengirimkan POST request ke server.
2. Pada pengujian digital *write*, pin yang digunakan adalah pin 5 yang dihubungkan pada *oscilloscope*.
3. Pada pengujian analog *write*, pin yang digunakan adalah pin 3 yang dihubungkan pada *oscilloscope* sedangkan duty cycle dari sinyal PWM yang digunakan adalah 25%, 50% dan 100%.
4. *Oscilloscope* yang digunakan adalah Tektronix TBS1000.

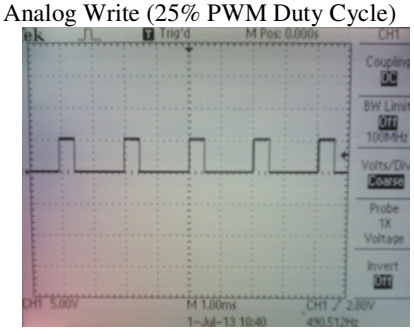
Hasil pengujian operasi *write* dapat dilihat sebagai berikut :

1. Digital Write (HIGH)



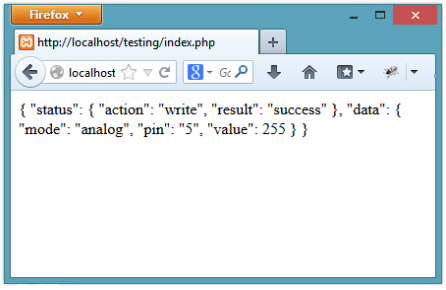
**Gambar 4.4 Hasil operasi digital write pada pin 5 (HIGH)**

2. Analog Write (25% PWM Duty Cycle)



**Gambar 4.5 Hasil operasi analog write pada pin 3 (25% PWM Duty Cycle)**

3. Data JSON (100% PWM Duty Cycle)




**Gambar 4.6 Data JSON hasil operasi write**

### 4.1.3 Pengujian *Delay*

Secara umum pengujian *delay* ditujukan untuk menguji performa dari sistem. Pengujian *delay* dibagi menjadi dua bagian yaitu pengujian dengan satu request dan pengujian

dengan banyak request. Pengujian dengn banyak request dilakukan dengan prosedur sebagai berikut :

1. Membuat sebuah *script* PHP dengan memanfaatkan modul cURL untuk mengirimkan 6 buah GET request secara bersamaan selama sepuluh kali. *Script* PHP tersebut dapat mencatat waktu total tiap request.
2. Request yang dikirimkan berisi permintaan untuk membaca nilai masukan pada pin A5.

   Hasil pengujian *delay* dapat dilihat sebagai berikut :

| Pengujian | Delay (s) |
|---|---|
| 1 | 0.014 |
| 2 | 0.014 |
| 3 | 0.014 |
| 4 | 0.016 |
| 5 | 0.014 |
| 6 | 0.014 |
| 7 | 0.015 |
| 8 | 0.015 |
| 9 | 0.016 |
| 10 | 0.016 |

**Tabel 4.1 Hasil pengujian delay dengan satu buah request**

| Uji | Request (Delay (s)) | | | | | |
|---|---|---|---|---|---|---|
| | I | II | III | IV | V | VI |
| 1 | 0.031 | 0.016 | 1.014 | 0.078 | 0.014 | 1.014 |
| 2 | 0.016 | 0.078 | 0.016 | 1.014 | 0.078 | 1.03 |
| 3 | 0.016 | 0.125 | 0.016 | 0.078 | 1.029 | 1.014 |
| 4 | 0.016 | 1.014 | 0.016 | 0.016 | 0.125 | 0.063 |
| 5 | 1.014 | 0.015 | 0.016 | 0.014 | 0.063 | 1.014 |
| 6 | 0.016 | 0.016 | 0.016 | 0.015 | 1.014 | 0.078 |
| 7 | 0.015 | 0.015 | 0.124 | 0.014 | 0.171 | 0.062 |
| 8 | 0.016 | 0.015 | 1.014 | 0.015 | 0.064 | 1.014 |
| 9 | 1.014 | 0.015 | 0.016 | 1.014 | 0.078 | 1.029 |
| 10 | 0.015 | 1.014 | 0.015 | 0.016 | 1.014 | 1.014 |

**Tabel 4.2 Hasil pengujian delay dengan enam buah request secara bersamaan**

### 4.2.1 Analisa *Delay*

Pada skripsi ini *delay* yang dicatat adalah *delay* total saat client mengirimkan request dan menerima response. Dari hasil pengujian, didapatkan data yang menunjukan *delay* total tersebut. Dari data tersebut dapat dihitung *delay* total rata-rata tiap-tiap request menggunakan persamaan sebagai berikut :

$$\bar{t} = \frac{\sum_{i=1}^{10} t_i}{10}$$

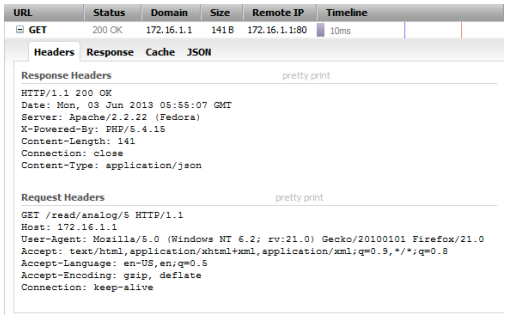
Pada pengujian delay dengan menggunakan satu request, delay total rata-rata adalah 14,8 ms sedangkan hasil perhitungan delay total rata-rata pada pengujian delay dengan enam request adalah sebagai berikut :

| Request | Delay total rata-rata (ms) |
|---|---|
| I | 216,9 |
| II | 232,3 |
| III | 226,3 |
| IV | 227,4 |
| V | 365 |
| VI | 733,2 |

**Tabel 4.3 Delay total rata-rata pada pengujian delay dengan enam request**

### 4.2.2 Analisa HTTP Request dan Response

Tujuan dari pengujian secara keseluruhan adalah melihat apakah sistem dapat bekerja sesuai dengan perancangan sebelumnya. Server dibuat dengan mengacu pada arsitektur REST. Pada pengujian sebelumnya, terdapat prosedur untuk menyimpan HTTP request dan response, data tersebut adalah sebagai berikut :




**Gambar 4.4 HTTP Request dan Response pada pengujian operasi *read***

## V. Penutup

### 5.1 Kesimpulan

Berdasarkan perancangan dan pengujian *Application Programming Interface Server* untuk Arduino maka dapat disimpulkan sebagai berikut :

1. *Application Programming Interface Server* untuk Arduino adalah suatu sistem yang memungkinkan sebuah aplikasi yang berjalan pada jaringan komputer dapat berinteraksi dengan arduino melalui *Application Programming Interface Server* dengan menggunakan protokol HTTP.
2. *Application Programming Interface Server* menyediakan metode untuk melakukan fungsi-fungsi manipulasi input dan output pada arduino.
3. Prinsip kerja *Application Programming Interface Server* untuk Arduino adalah menerima dan mengolah HTTP request dan menterjemahkan HTTP request menjadi TCP message yang akan dikirimkan ke arduino. Arduino akan menterjemahkan TCP message dari server

menjadi fungsi-fungsi manipulasi input dan output. Apabila fungsi tersebut telah berhasil dieksekusi, arduino akan mengirimkan TCP message ke server sebagai balasan. Server akan mengirimkan HTTP response dan data JSON yang sesuai dengan request dari client.

4. *Application Programming Interface Server* untuk Arduino hasil dari perancangan mampu menangani semua operasi manipulasi input dan output pada arduino yaitu *digitalRead, analogRead, digitalWrite* dan *analogWrite*.
5. *Delay* total rata-rata untuk mengirimkan satu buah request dan menerima response adalah 14,8 ms yang masih dalam batas wajar dan tidak akan terasa pengaruhnya oleh client.

### 5.2 Saran

1. *Class* ArduinoServer dapat dimodifikasi lagi sehingga dapat melakukan mekanisme *caching*, sehingga dapat membatasi request yang berulang-ulang pada arduino.
2. Menambahkan metode lain untuk dapat menggunakan *library* lainnya seperti *Servo* atau LCD, sehingga fungsi dari API Server menjadi semakin banyak.

## Daftar Pustaka


**[1]** RFC2616, Hypertext Transfer Protocol

**[2]** RFC793, Transmission Control Protocol

**[3]** Purbo, Onno W. 2001. *TCP/IP* – Standard, Desain, dan Implementasi. Jakarata: PT. Elex Media Komputindo

**[4]** Mansfield, Niall. 2003. Practical TCP/IP – Mendesain, Menggunakan dan Troubleshooting Jaringan. Jogjakarta: Andi

**[5]** Richardson, Leonard dan Sam Ruby. 2007. RESTful Web Services – Web Services for the real world. O'Reilly Media

**[6]** Masse, Mark. 2011. REST API Design Rulebook – Designing Consistent RESTful Web Service Interfaces. O'Reilly Media

**[7]** http://arduino.cc/en/Main/ArduinoEthernetShield diakses tanggal 25-02-2013

**[8]** http://arduino.cc/en/Reference/HomePage diakses tanggal 27-02-2013

**[9]** http://www.restapitutorial.com/ diakses pada 30-02-2013